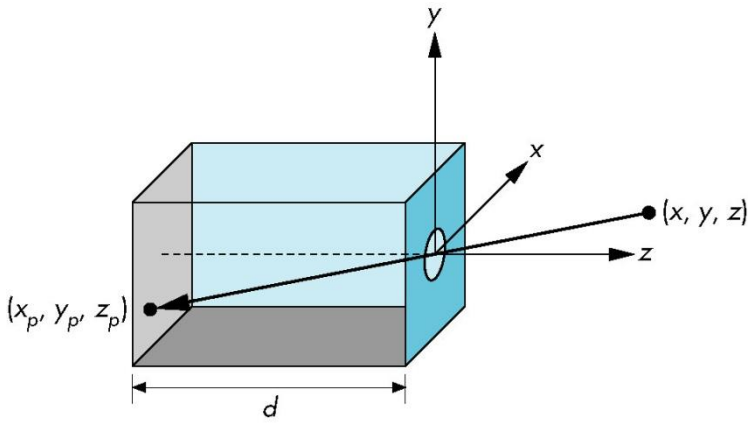
Lecture 2

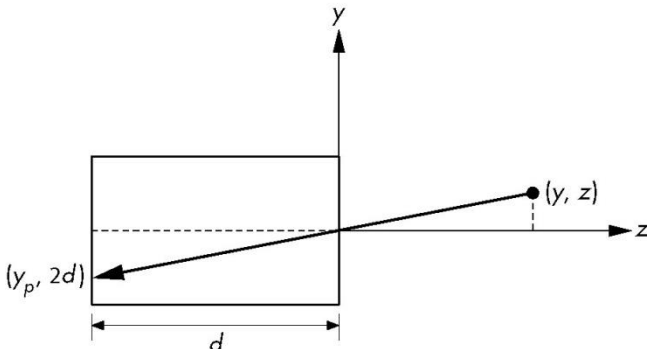
Imaging model and graphics system

In searching an imaging model (Position and projection)

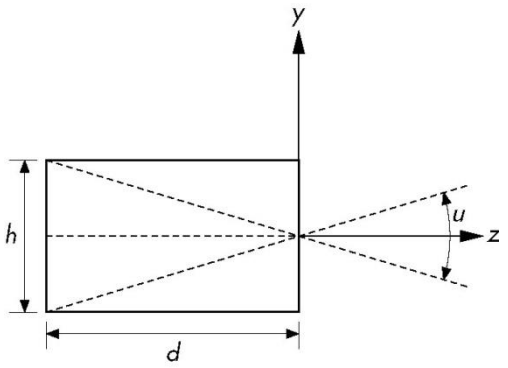


Pinhole camera

- Opaque box
- A small hole on one side
- A film on the hole facing side
- Distance between the two sides (focal length)



The camera is placed and oriented in the world (scene) right-hand coordinate system as shown
The point in the world at x, y ,z is imaged (projected) on the image plane (z=-d) as follows
Points outside the field of view is not projected

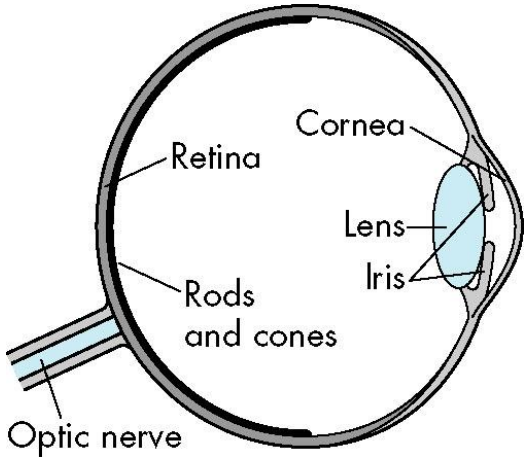


$$y_p = -\frac{y}{z/d} \qquad x_p = -\frac{x}{z/d} \qquad \theta = 2\tan^{-1}\frac{h}{2d}$$

In searching an imaging model (Colors, resolution)

Human visual system

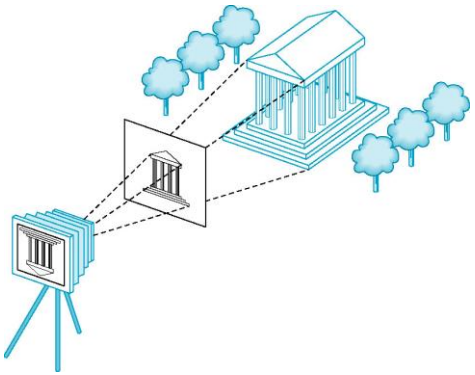
- Rods and cones are light sensors
- Rods works in low intensity light and don't sense different colors
- Cones need sufficient intensity to work
- There are three types of cones each type responds to a specific range of light frequencies (color)
- The number of cones and rods determines the visual acuity

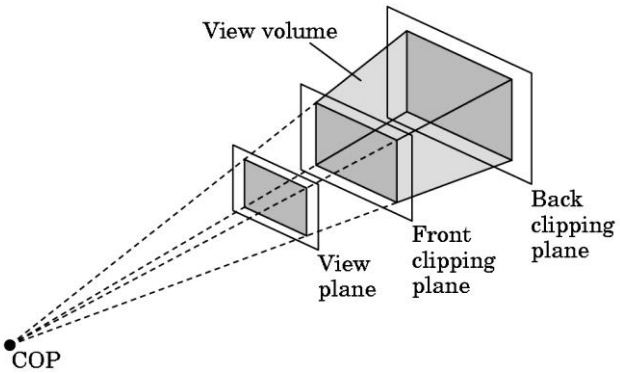


Imaging model

- Projection provides us with location of a point in the scene as a point the image
- The point in the image should have cones and rods models (RGB, and brightness)
- We should have a shading model that emulates how the cones are affected to calculate the RGB and brightness components

Computer graphics Imaging Model (Projection: the synthetic-Camera model)



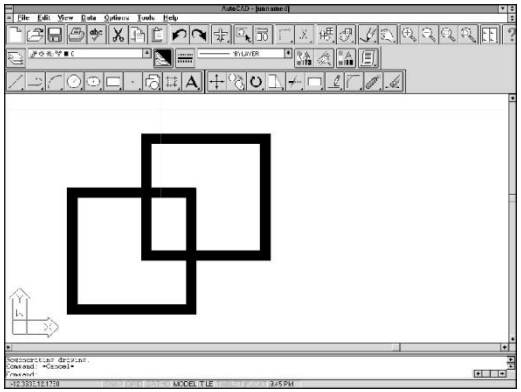


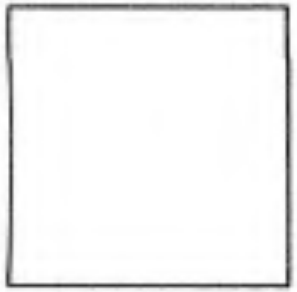
Instead of exactly imitating the pinhole camera projection we can imitate synthetic camera that gives us more freedom

- The projection plane is placed in front of the pin hole. This give a not inverted image.
- The projection of a point in the world is the intersection of the line from the point to the center of the hole (center of projection COP). This line is called the projector
- In the synthetic-Camera, we are not limited by the physical film area. Instead, we specify a clipping window (clipping rectangle): the area that will be considered as the image in the projection plane
- The pinhole location is the view point
- The line from the view point to the center of the clipping window in the view direction
- We can specify a far clipping plane to specify the frustum which it's contents is imaged

Programmer interface: Pin plotter model

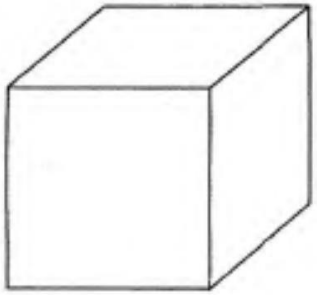
Pin plotter conceptual model: The programmer (graphics creator) works in 2D to create 2D and 3D graphics using function calls or user interfaces. Hence, He works out the required projection, shading, hidden surface removal, etc., to reduce everything to 2D objects





```
moveto(0, 0);
lineto(1, 0);
lineto(1, 1);
lineto(0, 1);
lineto(0,0);
```

Add



```
moveto(0, 1);
lineto(0.5, 1.866);
lineto(1.5, 1.866);
lineto(1.5, 0.866);
```

Programmer interface: 3D Graphics API

3D Graphics system conceptual model:

- The programmer (graphics creator) works directly in the domain of his problem to create 2D and 3D graphics using function calls
- He specify objects, shading methods, the camera(viewer) specification, object material specification
- The Graphics system (library and GPU) does the projection, shading, etc to produce the image.
- One of the most famous graphics library is the OpenGl library
- Another famous 3D graphics library is the Direct3D (MS)

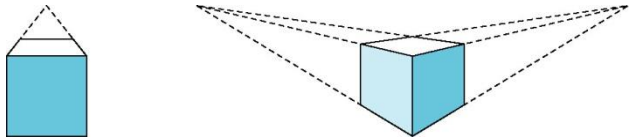
We can use a 3D graphics system as a 2D system. Just put the viewer behind the xy plane in the world coordinate and make the xy plane as the projection plane. Then continue drawing in 2D by setting any z to zero

Report Discussion

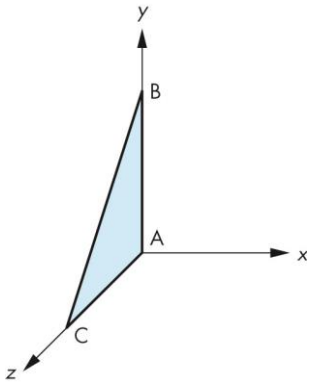
Previous lecture repot:
None

Next lecture repot:
Read below

Consider the perspective views of the cube shown below. The one on the left is called a one-point perspective because parallel lines in one direction of the cube—along the sides of the top—converge to a vanishing point in the image. In contrast, the image on the right is a two-point perspective. Characterize the particular relationship between the viewer, or a simple camera, and the cube that determines why one is a two-point perspective and the other is a one-point perspective



Object specification



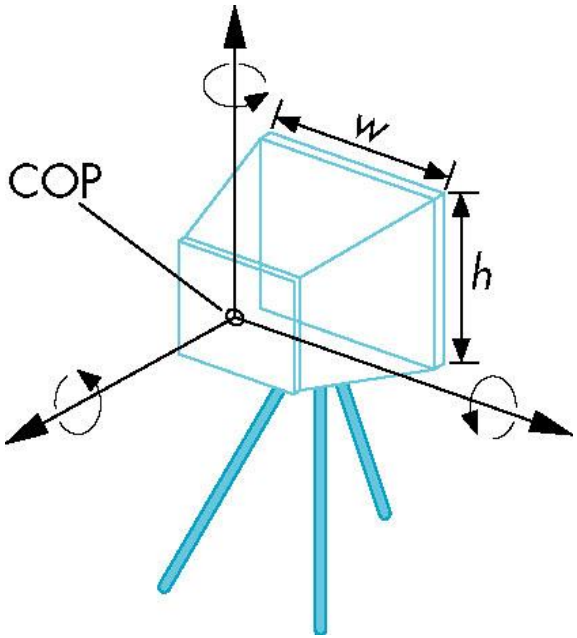
type of object

```
glBegin(GL_POLYGON);
    glVertex3f(0.0, 0.0, 0.0); /* vertex A */
    glVertex3f(0.0, 1.0, 0.0); /* vertex B */
    glVertex3f(0.0, 0.0, 1.0); /* vertex C */
glEnd( );
```

End of objet

- Most graphics library provides functions to specify certain set of primitive objects (those that can be displayed and processed rapidly by the hardware)
  - Each of these primitives is usually completely specified by a set of points (vertices)
  - Point : one vertex
  - Line segment: Two vertices
  - Polygon: number of vertices
- More complex objects are usually specified using these primitives (either by the programmer or by the library)
- OpenGl provides function to specify curves, surfaces and to work directly on pixels in the FB

Camera or viewer specification



Camera or viewer specification:

- Position: specifies the COP point in the world
- Orientation: we can rotate the camera around any of the three axis of a coordinate system centered at the COP (camera coordinate system)
- Focal length: distance from COP to the image plane along the view direction
- Image plane: The size of the world (frustum) images is specified by the focal length and the size of the image plane(w , h)
- Some API allow adjusting the orientation of the image plane independently from the viewer orientation
- The contents of the image plane could be mapped to the whole drawing winnow or to a view port (part of the window)

Camera or viewer specification continued

In OpenGl and most other API, camera or viewer specification can be set in various ways :

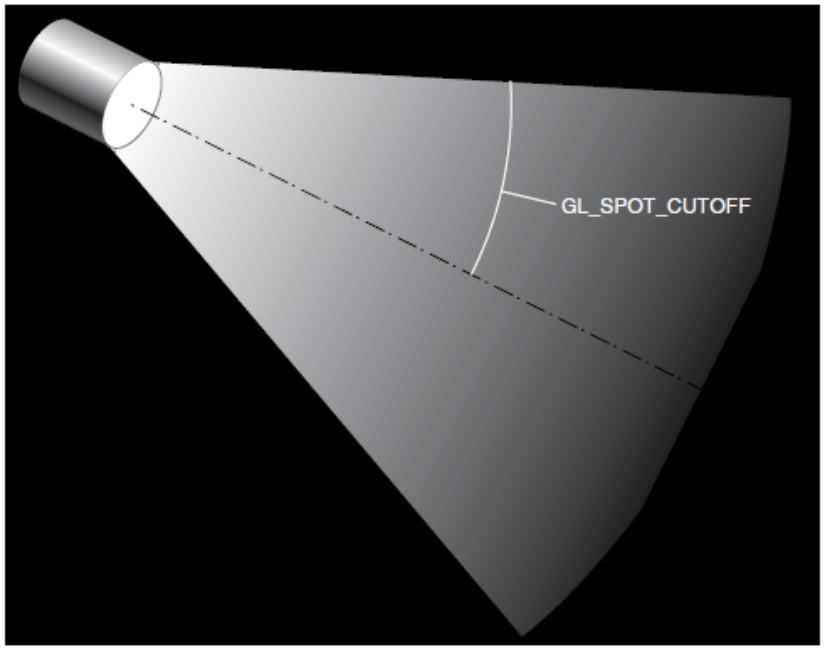
- Transformation matrices (to discuss later)
- Positional, size, orientation parameters

```
gluLookAt(cop_x, cop_y, cop_z, at_x, at_y, at_z, up_x, up_y, up_z);
glPerspective(field_of_view, aspect_ratio, near, far);
```

- First call:
  - Cop: center of projection location in the world
  - The view direction is from cop to at points in the world
  - The up is unit vector direction that specifies the rotation of the camera around the view axis
- Second call: specifies the frustum and the projection

Light specification

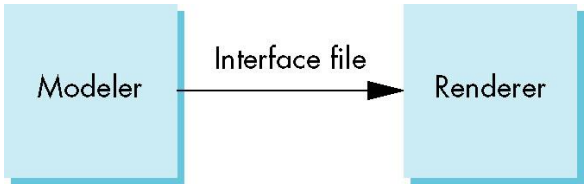
- A light source is defined by position
  - Type: emits in all direction, in a cone shape, etc.
  - Light characteristics: Color, intensities, etc
- Many light sources to lighten the same scene




```
GLfloat light_position[] = { 1.0, 1.0, 1.0, 0.0 };
glLightfv(GL_LIGHT0, GL_POSITION, light_position);

GLfloat spot_direction[] = { -1.0, -1.0, 0.0 };
glLightfv(GL_LIGHT0, GL_SPOT_DIRECTION, spot_direction);
```

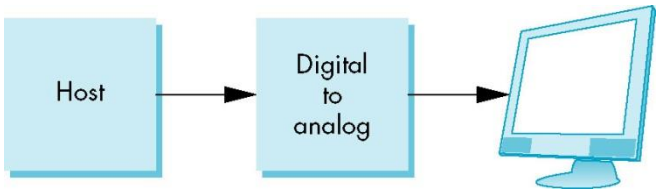
The modeling -rendering paradigm



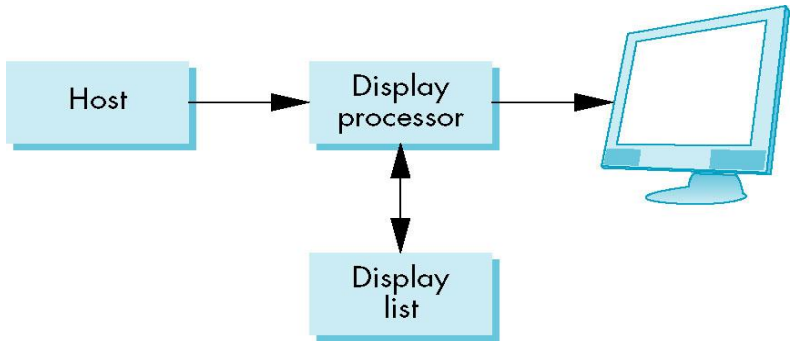
- Modeling
  - Interactive
  - Synthetic camera projection is enough to give interactivity to designers
  - Model data include objects, lights, and camera specification
- Rendering
  - Not interactive
  - Computation intensive

Graphics architecture

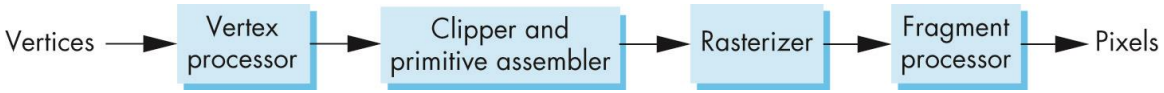
- In early graphics architectures, the main processor was responsible to display and refresh images!!!!
- All the required work starting from the specification of the objects in terms of vertices to producing the image pixels in the buffer can be sequenced in processing stages in a pipeline (this is the way modern GPU is implemented)
- With the pipeline architecture, more processing could be included without affecting the throughput of the system while increasing the latency
- Each stage in the pipeline may be configurable by parameters or completely programmable



Early graphics systems



Display-processor architecture



Graphics pipeline